**Claims:**

What is claimed is:

1. A system for software source code analysis, comprising:

a software interface to a source code management system, which can be used to identify changes that have occurred to source code over a specified interval of time or relative changes, independent of any particular source code management tool implementation;

a software interface to a code coverage database, which can be used to identify what source code has been exercised during a test run, independent of any particular code coverage tool implementation;

a software interface for identifying which source files are exercised in a software product by a particular software test, using code coverage mapping,

a software interface for identifying what tests have failed during a test execution cycle for a particular software product, independent of any particular testing technology or test execution tool; and

a bug inspection analyzer that determines the failure-to-change intersection point by integrating the information culled from the software interfaces described above to determine what tests failed, what source code files of the target software product are exercised by the failing tests, and which of the files identified thereby have been changed by a product software developer since the failed tests were last known to be in a passing state or within some other specified timeframe.

2. The system of claim 1 wherein changes to the software code between said first version and said second version are identified by change label.

3. The system of claim 1 wherein changes to the software code between said first version and said second version are identified by modification date.

4. The system of claim 1 wherein the code coverage interface includes an input for allowing an operator to specify either of date and/or code change ranges to be analyzed by said bug inspection analyzer.

5. The system of claim 1 wherein the interfaces are realized to interact with a tool-specific implementation that interfaces to a vendor-specific subsystem.

6. The system of claim 1 wherein the source code interface includes an interface to a vendor-specific SCM system.

7. The system of claim 6 wherein wherein the system includes said SCM system.

8. The system of claim 1 wherein the test interface includes an interface to a TER system.

9. The system of claim 8 wherein wherein the system includes said TER system.

10. The system of claim 1 wherein the test interface includes an interface to a code testing system.

11. The system of claim 8 wherein wherein the system includes said code testing system.

12. A system for software source code analysis, comprising:

means for retrieving a software code and running test suites against it at a first time and at a second time;

means for importing code coverage data into the framework for failure

analysis;

means for retrieving detailed set of line-level product changes from a source code management system; and,

means for comparing line-level code coverage data for a test case from a code coverage toolset to line-level change information from a source code management system, and determining an intersection of these two data sets to represents the set of critical changes over the specified time period.

13. A method for software source code analysis, comprising the steps of:

accessing a source code management system, which can be used to identify changes that have occurred to source code over a specified interval of time or relative changes, independent of any particular source code management tool implementation;

accessing a code coverage database, which can be used to identify what source code has been exercised during a test run, independent of any particular code coverage tool implementation;

identifying which source files are exercised in a software product by a particular software test, using code coverage mapping;

identifying what tests have failed during a test execution cycle for a particular software product, independent of any particular testing technology or test execution tool; and

determining the failure-to-change intersection point by integrating the information culled from the software interfaces described above to determine what tests failed, what source code files of the target software product are exercised by the failing tests, and which of the files identified thereby have been changed by a product software developer since the failed tests were last known to be in a passing state or within some other specified timeframe.

14. The system of claim 13 wherein changes to the software code between

said first version and said second version are identified by change label.

15. The system of claim 13 wherein changes to the software code between said first version and said second version are identified by modification date.

16. The system of claim 13 wherein the code coverage interface includes an input for allowing an operator to specify either of date and/or code change ranges to be analyzed by said bug inspection analyzer.

17. The system of claim 13 wherein the interfaces are realized to interact with a tool-specific implementation that interfaces to a vendor-specific subsystem.

18. The system of claim 13 wherein the source code interface includes an interface to a vendor-specific SCM system.

19. The system of claim 18 wherein wherein the system includes said SCM system.

20. The system of claim 13 wherein the test interface includes an interface to a TER system.

21. The system of claim 20 wherein wherein the system includes said TER system.

22. The system of claim 13 wherein the test interface includes an interface to a code testing system.

23. The system of claim 20 wherein wherein the system includes said code testing system.

24. A method of software source code analysis, comprising the steps of:

retrieving a software code and running test suites against it at a first time and at a second time;

importing code coverage data into the framework for failure analysis;

retrieving detailed set of line-level product changes from a source code management system; and

comparing line-level code coverage data for a test case from a code coverage toolset to line-level change information from a source code management system, and determining an intersection of these two data sets to represents the set of critical changes over the specified time period.

25. A computer readable medium including instruction stored thereon which when executed cause the computer to perform the steps of:

accessing a source code management system, which can be used to identify changes that have occurred to source code over a specified interval of time or relative changes, independent of any particular source code management tool implementation;

accessing a code coverage database, which can be used to identify what source code has been exercised during a test run, independent of any particular code coverage tool implementation;

identifying which source files are exercised in a software product by a particular software test, using code coverage mapping;

identifying what tests have failed during a test execution cycle for a particular software product, independent of any particular testing technology or test execution tool; and

determining the failure-to-change intersection point by integrating the information culled from the software interfaces described above to determine what tests failed, what source code files of the target software product are exercised by the failing tests, and which of the files identified thereby have been

changed by a product software developer since the failed tests were last known to be in a passing state or within some other specified timeframe.